UNDERSTANDING THE HLA INTERFACE (A JOB FOR THE INTELLIGENCE ANALYST ?)

Duncan Clark
Data Sciences (UK) Ltd
Meudon House, Meudon Avenue,
Farnborough, Hants,
GU14 7NB, UK
D.G.Clark@datasci.co.uk
Peter Hoare
DRA Malvern
E211, DRA Malvern, St Andrews Road,
Malvern, Worcs,
WR14 3PS, UK
peteh@signal.dra.hmg.gb

KEYWORDS
HLA Interface, Practices

ABSTRACT

This paper describes the authors' attempts to understand the HLA Interface with a view to assessing how an existing simulation and modelling framework could support HLA federation development.

The exercise has been undertaken outside the main HLA activities and has therefore relied on the publicly available interface specifications. While the interface was often quite specific, the information available for assessing how a federate should operate was often vague or open to misinterpretation. This latter information had to be inferred by piecing together bits of information from different parts of the specification using similar techniques to that of an Intelligence Analyst. The authors contend that many issues related to the design of federations are outside the scope of the HLA interface specification, but are essential to the interoperability of HLA federates.

In order to fully understand the Interface, an analysis has been undertaken of the RTI and its role in a federation (using the Object Modelling Technique notation). This identified the expected behaviour of a federate as well as concisely documenting the response of the RTI. Having been through this exercise, the authors maintain that the interface specification should include documentation to illustrate how the interface should be interpreted and how federates could achieve the benefits that HLA offers.

1. PROJECT OVERVIEW AND OBJECTIVES

1.1 The existing Framework

Data Sciences have been developing Analytical models for many years to support Operational Analysis, COIEA and Engineering studies mostly in support of UK MOD activities. Three years ago they decided to develop a simulation and modelling framework and library to support these activities. This was the Library for Object Oriented Modelling (LOOM).

The initiative was based on Object Oriented Technology with similar high level objectives (on a smaller scale) to those being sought with HLA. Since its conception about 10 models have been successfully produced, each with a high degree of reuse of the central framework and concepts of operation, yet each serving a very different application.

The DIS architecture has been around for some time, but was not seen as suitable for the constructive simulations supported by LOOM. The High Level Architecture, however, seemed to complement the concepts within LOOM and provide scalability as well as improved interoperability.

The objective was therefore set to examine how compliant the LOOM concepts were with HLA and how LOOM would support new simulations that wished to interoperate using HLA.

1.2 Understanding the HLA

The first stage of the study was to understand the High Level Architecture. Information was obtained from the Internet mainly from the DMSO site[1], but also the DIS mail reflectors (conferences).

Slide presentations from early briefings provided an outline of how HLA was intended to work. The high level concepts being proposed seemed easy to understand, pragmatic and convincing in how they would achieve the objectives being set for HLA.

The draft Interface Specification[2], Object Model Template specification[3] and API also became available on the Web soon after the start of the project. This was the sole basis of our Analysts' "Intelligence information".

From this information we had to build a model of how the RTI interface **could** work. We then tested how a federate (or federation) would interact with the RTI if it did work that way. When problems arose we iterated round the loop again, changing either our model of the RTI or the way the federates interacted with it. When faced with the issue of not knowing something about the RTI, the Intelligence Analysts' premise of "How would I have done it?" was applied.

While this was going on the RTI was still under development. Many of our assumptions about how the RTI would or wouldn't work were proven to be correct when the next release of the specification appeared showing a feature already included in our model.

1.3 Framework HLA Compliancy Assessment

In the early stages of the project we (naively) believed that if we could identify the key interface requirements being specified for HLA, we could produce a matrix indicating which of those requirements LOOM (or any other federate) could meet. Even if (as expected) some of the requirements would not be met immediately by the "candidate" simulation, the ease with which they could be incorporated could be assessed.

In practice we found that the HLA interface could be interpreted in so many ways that it was very hard to identify a set of interface "requirements" against which a simulation could be assessed in any way. In effect we needed a federation interface specification as well as an HLA interface specification before we could really look at any form of compliancy assessment.

In order to come up with a set of "federation" requirements we looked at the typical simulations that were already undertaken by LOOM, to see what they had to do. Using our Intelligence Analysts assumption that an HLA federation might do the same thing we then postulated how the HLA Interface would be used.

2. KEY ISSUES IN INTERPRETING THE HLA SPECIFICATION

This section describes the main issues that the authors had to address in order to understand the HLA interface.

2.1 Simulation / Federation Management

In constructive analytical simulations, the current main use of distributed processing is for speeding up the simulation run. A distributed simulation framework also offers advantages in interoperability between simulations. In either of these cases, there must be some control over which simulations are actually running and how the workload is allocated to different simulations.

As well as management of simulations joining and leaving, there is usually a "control" within a constructive simulation that manages the pausing / resuming of simulated time, and saving / restoring of simulation state.

Initially HLA seemed to provide no guidelines for this control but about the time we were grappling with the issue, the Interface Specification started to include the "Federate (Manager)" as an initiator for interfaces. From this we concluded that the HLA team had agreed that "all simulations are equal but some are more equal than others".

2.2 FOM / SOM consistency

Having agreed the responsibilities of the simulations in Federation Management, we then applied our Intelligence Analyst approach to the problem of how the different simulations would understand each other. This now addressed the modelling domain and the common language used to describe the objects and attributes that each simulation understands: the Object Model Template.

The basic features of inheritance and attributes seemed straightforward until we tried to work out how each simulation worked out which integer (according to the API) corresponded to which ObjectClass and/or attribute. There either had to be a compiled map that must be used by each of the simulations, or the RTI had to provide a Name to Integer (and vice-versa) mapping service based on the loaded FOM. The next release of the API contained mapping service calls.

The second area of concern was that of "aggregation". The tables in the OMT show an aggregation facility (although this is now "optional"). Early examples were very confusing about how this table should be interpreted, especially when inheritance was concerned and when we found the structure of the RID, there was no way the RTI could take any notice of the tables. The implication was that each SubObject had to be created individually; it was not possible to create a ship, for example, and have all the systems on that ship created automatically.

2.3 Scenario Management

Many DIS based simulations are explicit about the objects they represent and there is no real opportunity for any "scenario management" to be exercised. In analytical applications it is vitally important that a scenario be controlled, and hence repeatable to ensure that the results can be fully understood. The HLA seemed to offer no rules on how this should be achieved although the attribute transfer facilities that the HLA offers for scenario management appear very powerful if they could be used effectively.

The model proposed was that some federate (probably a federate manager) would undertake the bulk of the initial object creation from a scenario database that was under configuration management.

The objects identified could then be claimed or allocated (for representation) using the ownership management mechanisms. This was also seen as having great potential for changing between levels of fidelity in the models themselves.

2.4 Time Management

Many simulations have their own preferences for time management mechanisms and the communities involved with the HLA appear to divide up into those using real time control (as in DIS) and those that need to retain causality in the timing of events occurring and being interpreted.

Early versions of the interface specification did not indicate how causality would be maintained over a distributed system nor what would be expected of a federate in order to operate in this manner.

The Time Management Papers[4] provide clear overviews of how both time management systems would operate. However the API seems to have lagged behind, implying that the RTI might not fully support all these mechanisms.

3. HLA COMPLIANCY ASSESSMENT

This section identifies the way the authors looked for HLA compliancy in the LOOM framework.

3.1 Model Class and Attributes Structure Mapping Capability

One of the key rules of the HLA is the ability to agree to a common Federation Object Model. This implies that a federate should have a mapping facility between its own model objects and attributes and the indexing systems used within the HLA.

This was provided in LOOM by a concept named "Model Characterisation" which provides complete descriptions of each class's attributes, relationships and inheritance hierarchies. This is used primarily to support graphical data entry and display and reporting facilities but could easily be extended to support the publish and subscribe interfaces.

3.2 Object Management capability

A key issue in the HLA is the ability to dynamically respond to objects appearing in and disappearing from the scenario. Similarly the HLA requires that simulations need to be able to take over, or allow

another model to take over, individual object attribute modelling.

To check compliancy in this area we looked for mechanisms within LOOM that would allow "objects" to be created according to specified generic types. Model Characterisation would allow each of these to have different data driven behaviour and indeed different structures. The granularity of the model structure would need to be finer than the Object classes since each attribute that could be transferred would need to be derived in a separate entity to allow it to be separated out from the main object processing and updated by an external model.

3.3 Control of Attribute Updates and Interactions

Any simulation will update the internal variables representing the modelled object attributes in response to its interactions with other modelled objects.

For compliancy we were looking for a simple way in which the model could inform the outside world that the attribute has been changed (and vice-versa). LOOM offers a facility for registering interests in characterised attributes, allowing the model (or the RTI) to be informed whenever that attribute is altered. This provides an effective manner in which attribute updates can be supported.

Interactions imply some sort of event notification and processing in the candidate federate. The LOOM framework supports the concept of critical events. These can be raised by a model (on detecting an interaction) and (like the attributes) cause a registered routine to be run when they occur and hence invoke the send_interaction call. Similarly a receive_interaction can raise a critical event when it is called and thus inform other models within the LOOM simulation.

3.4 Time management compatibility

Most of the simulations supported by the LOOM framework are not real time but do need to maintain a strict sequence of events. In order to assess time management compliancy we needed to decide what scheme we ought to be using. This would not be real time so some form of causal time management should be sought.

Since there was originally no support for distributed simulation in the LOOM framework, the concept of lookahead was not catered for. While zero lookahead schemes would probably work this would not operate very efficiently in a distributed environment. Other mechanisms for obtaining a lookahead were also investigated such as examining the event queue.

4. THE HLA INTERFACE SPECIFICATION MODEL

4.1 The HLA Object Model

The first stage in analysing the interface was to try to visualise the structure of the key objects within a federation. This was undertaken using a tool called Select OMT Professional, which follows the Object Modelling Technique method by Rumbaugh et al [5]. This object model is shown in figure1. Note that although the Federation and RTI are shown in the diagram they do not really exist as real objects. The diagram just attempts to indicate which classes are considered to be in these groupings.

On the diagram the boxes represent classes and lines represent relationships with arrows indicating direction of association. Inheritance is indicated by the triangle symbol, while aggregation is indicated by the diamond symbol on the composite class. Multiplicity is indicated with a circle at the end of a relationship.

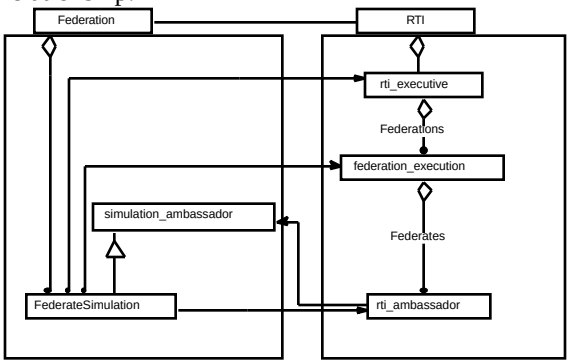


Figure 1 HLA Object Model

Having identified these key objects and how they fit together, the Interface specification was then examined together with the API, to clearly define the operations that were considered part of the interface, and to map those onto the RTI object classes and the simulation_ambassador as shown in figure 2 for the federation_execution class.

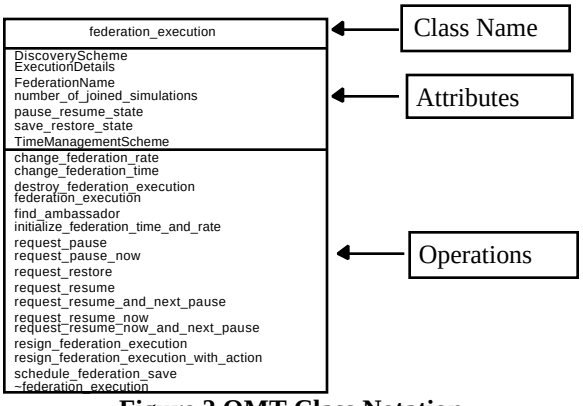


Figure 2 OMT Class Notation

By including the interface specification documentation in the database, together with the Interface Definition Language (IDL) in the API, and more recently C++ operation signatures, a readily maintainable, compact definition of the interface is obtained. This can be displayed within the class on the diagram giving a very quick overview of the actual operations and attributes of the class. The detailed information is maintained in the database and may be accessed directly by clicking on the method, or generated in reports. This gave us a model with which we could start to "experiment", to find out how the RTI could operate and be used.

4.2 The Federation Object Model

The other important aspect of the HLA interface is the Object Model Template. Although this is specified in table form, the concepts lend themselves to representation in the Select Database as shown in figure 3.

Building this structure up also identifies the data structures that may be used within the DFOM or read in through the RID. However the format of the RID indicates that not all this information is actually read in and hence implies constraints in the way that the RTI operates.

First, there is no mapping between the ID used in the RTI to identify FOM Object types (ObjectClass, AttributeName, etc.) and the actual name (Image) used in the FOM and RID. While there are now access procedures to convert between them, two independent federates must either both use these or else use the same RID reader and same file to generate an identical map within themselves.

Second, the Component Structure Table does appear in the RID. This implies that if any composite object is created, the subobjects must also be created by the federate: they do not get created automatically.

Very few of the Parameter and Attribute fields are used. It must be assumed that simulations "hard code" this information (especially type information) into their interfaces. This would imply that receiving simulations should check that the supplied information does adhere to the "agreed" field values.

It is noted that the SOMs now have an option for indicating Publish and Subscribe capability. While this information is important for the design of a federation, it appears to duplicate the information passed at runtime with class based Object declaration calls.

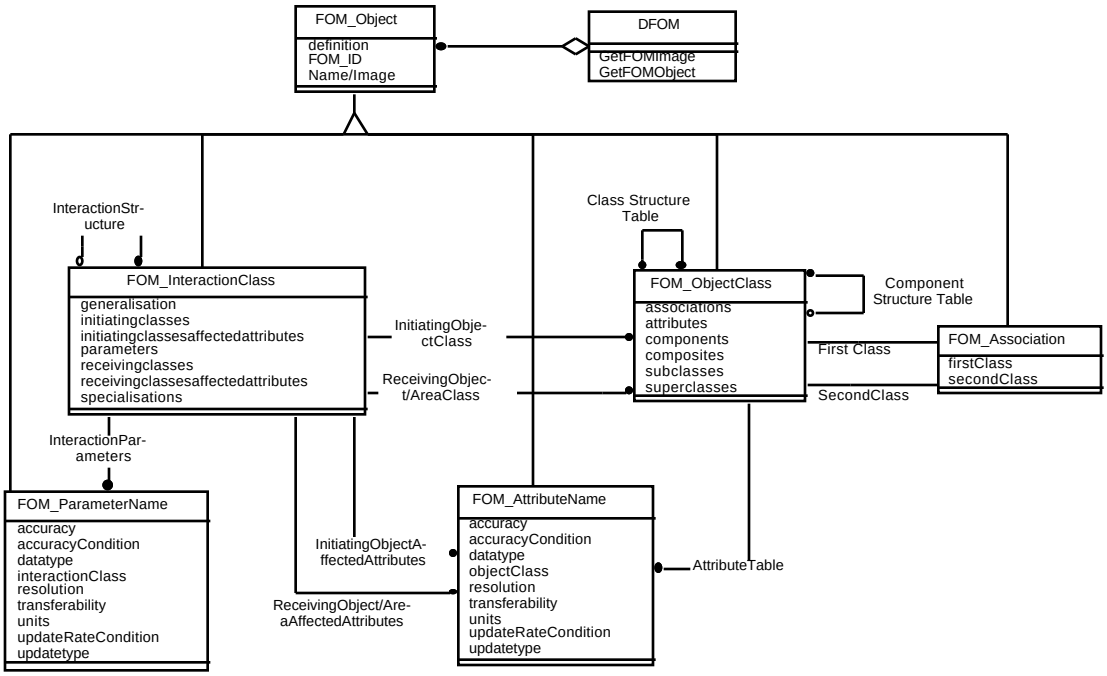


Figure 3 Object Model Template Object Diagram

4.3 The RTI expanded Model

Through this modelling a better picture evolved of what information the RTI was actually passing through its interfaces. This led to assumptions about what the RTI would do with the information and how it would maintain its “databases”. While the actual implementation of the RTI is likely to be complex due to its distributed nature, the logical data structures needed to support the databases and the RTI operations can be quite simple, as shown overleaf.

By treating the RTI as a monolithic object, the availability of information can be traced, which in turn provides clues as to the sequences in which calls may be made. The results of this investigation are shown in the Use Case diagrams in the next section.

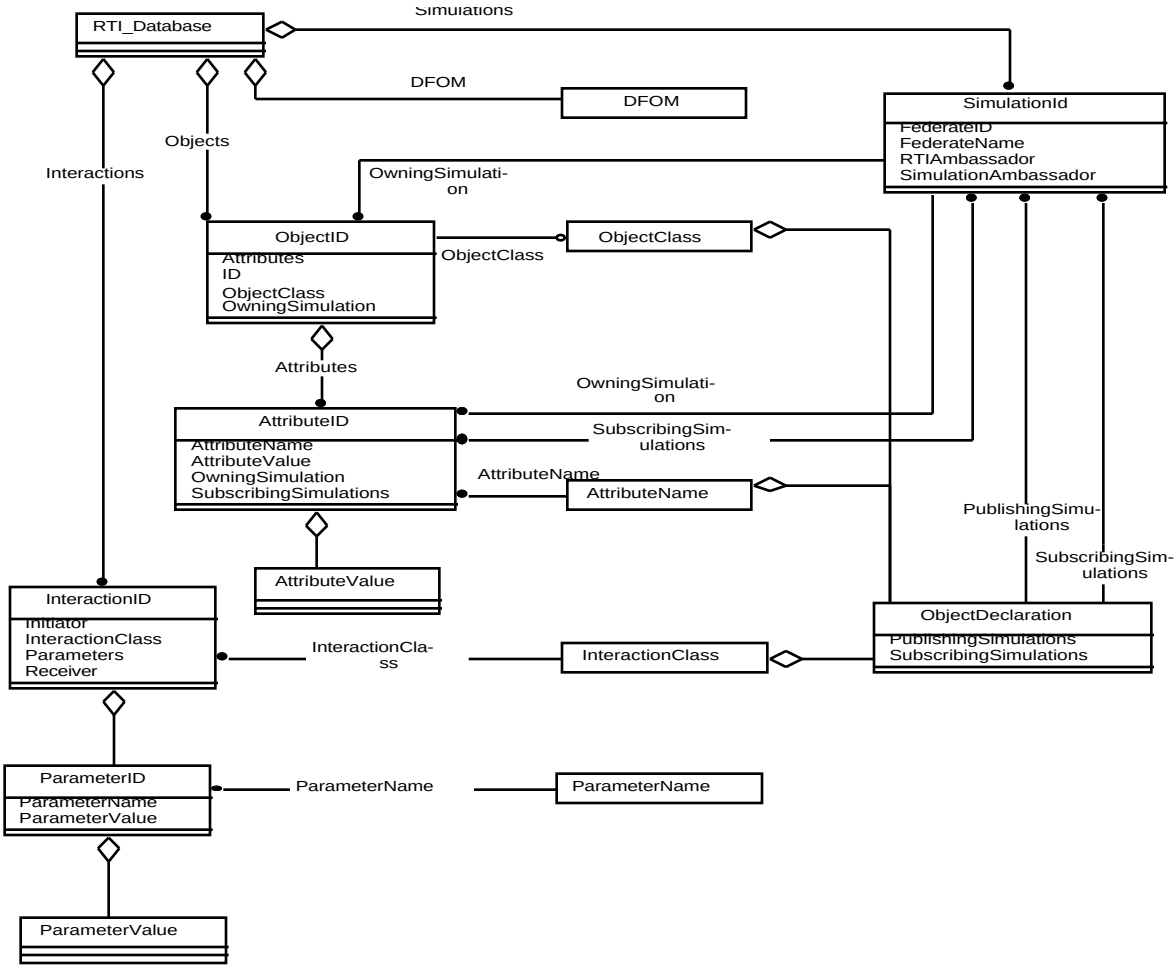


Figure 4 RTI Database Object Model

5. THE FEDERATION USE CASE SCENARIOS

Having built a model of the RTI operations the next stage in understanding the model was to “simulate” the calls that a federate should make to the RTI and what would happen as a result of those calls. This “simulation” also investigated what was expected of a federate, when called, in order to achieve the objective of the RTI.

Each of the main areas of the interface were simulated with a Use case diagram as shown in the following subsections. (Different ways of using the RTI could be expanded into more Use cases).

The Use case diagrams show the interactions between objects (each arrow is a call to the method indicated on the object pointed to). Time sequences go down the page. Classes or objects to the left of the vertical line are considered outside the RTI system boundary. Calls back to the federate occur though the simulation_ambassador on the right hand side, but continue from the federate itself on the left hand side.

There is no commentary on the Use Cases, since the authors believe that they should tell their own story. It is important to point out that this is a current understanding of what should occur and as work progresses and the model becomes more developed, these diagrams will be updated.

5.1 Federation Management

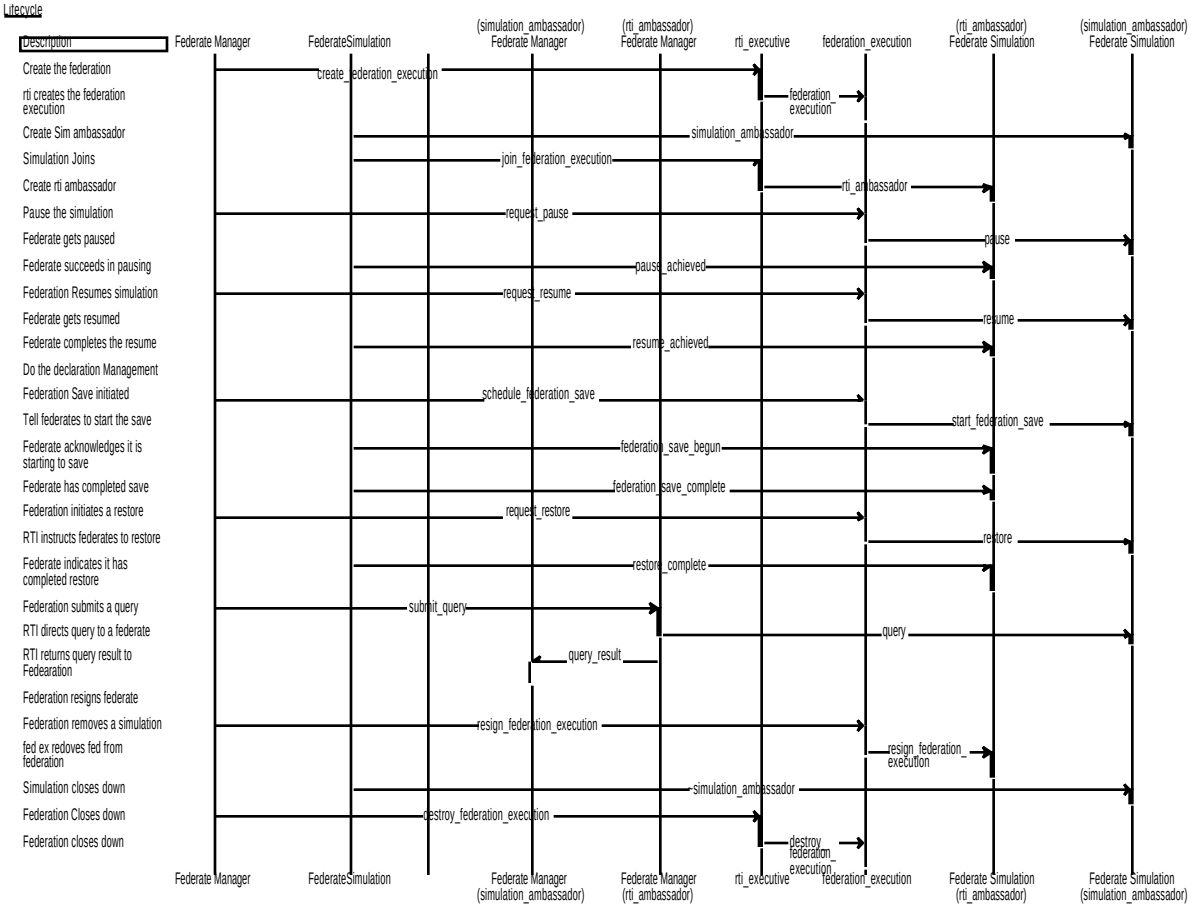
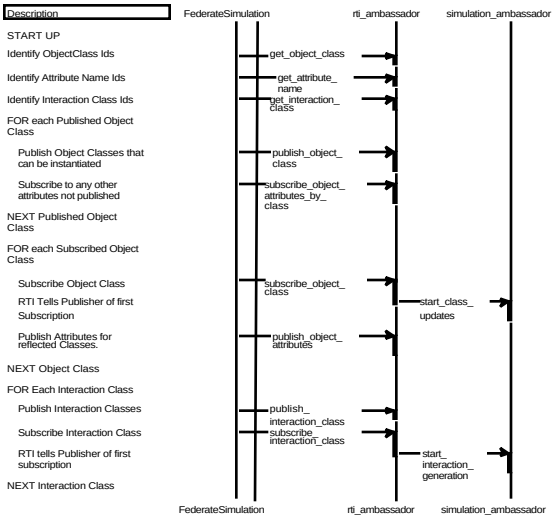


Figure 5 Federation Management Use Case

5.2 Declaration Management

DeclarationManagement



DeclarationManagement

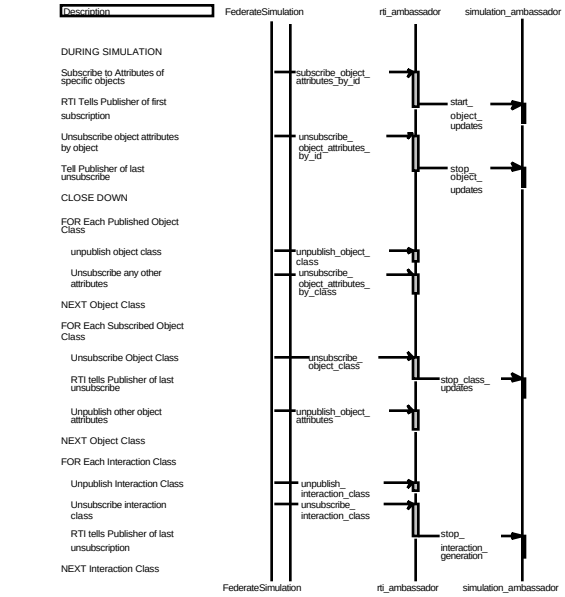
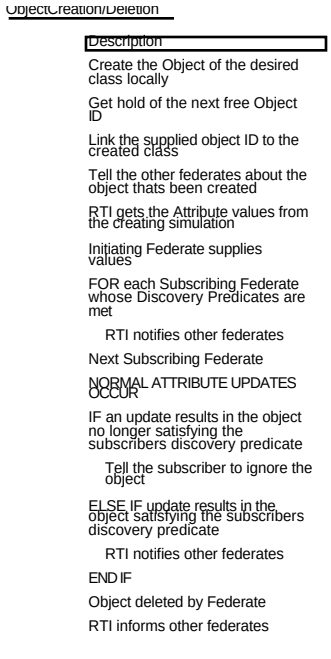


Figure 6 Declaration Management Use Case

5.3 Object Management - Creation and Deletion



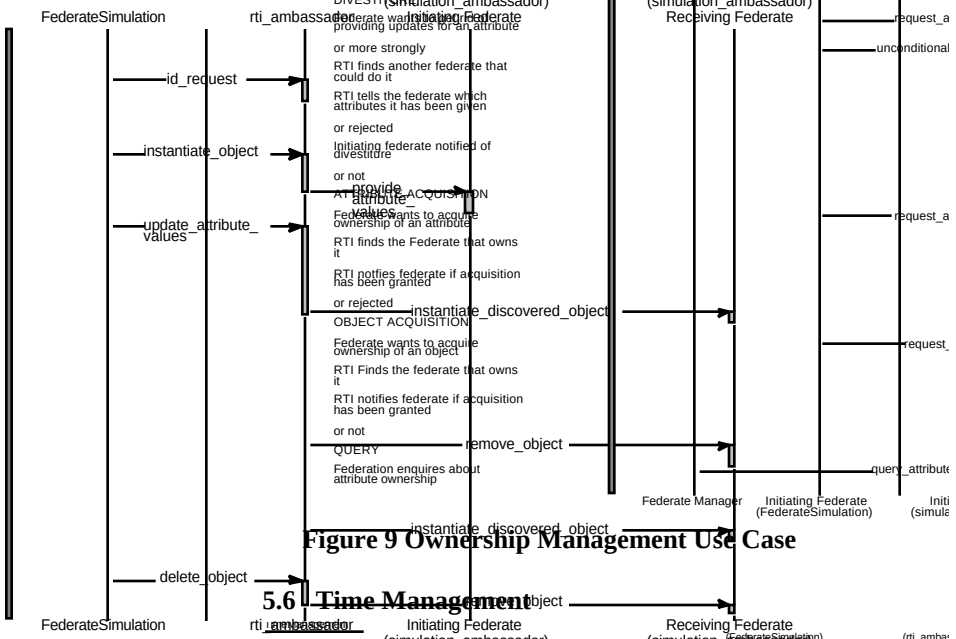


Figure 7 Object Creation and Deletion Use Case

5.4 Object Management - Control of Updates

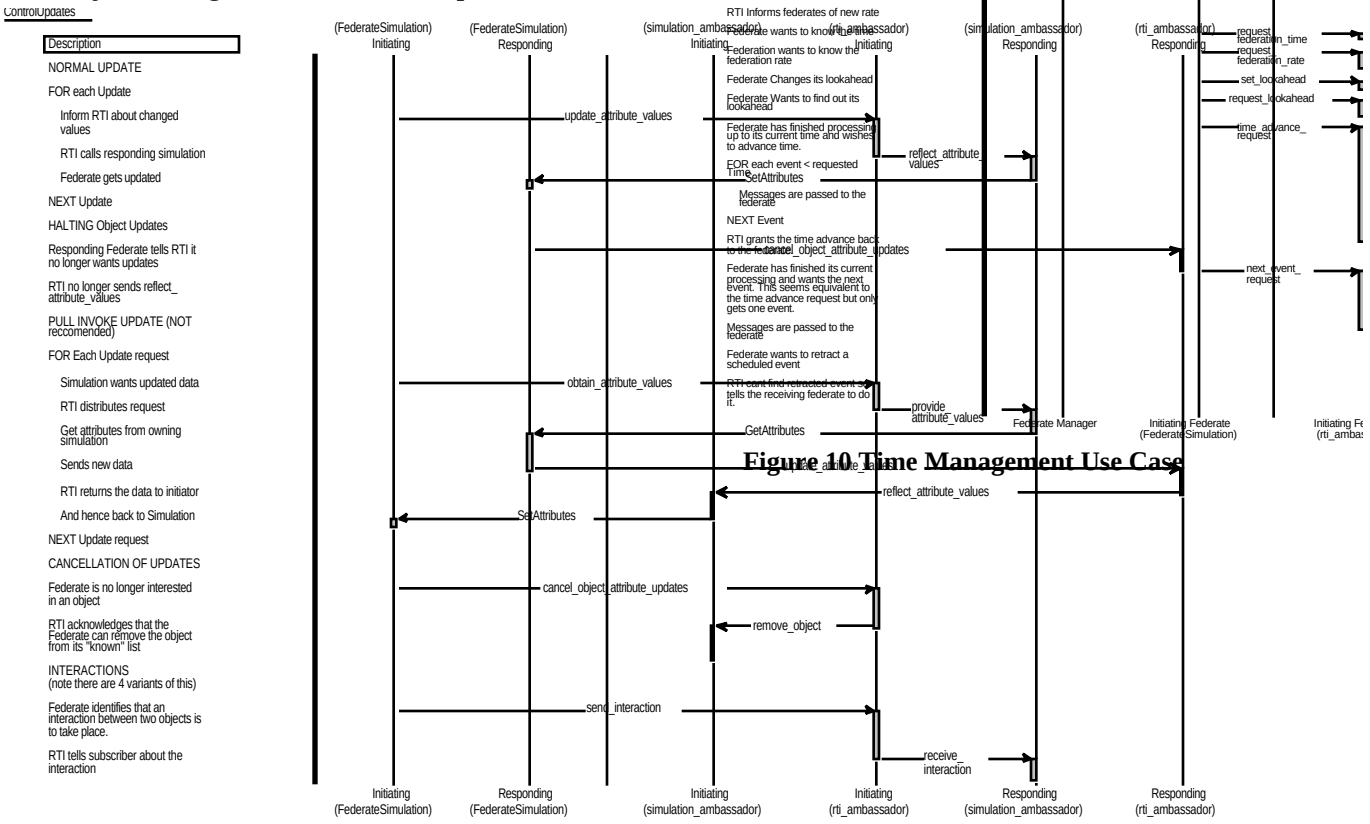


Figure 8 Attribute Update and Interaction Use Case

5.5 Ownership Management

Figure 9 Ownership Management Use Case

5.6 Time Management

Figure 10 Time Management Use Case

6. CONCLUSIONS

6.1 Recommendations for Federate HLA compliancy assessment

The original objective of the study was to examine how compliant LOOM based simulations were with the HLA interface. The easy answer to this is: it must obey all the rules, implement the simulation_ambassador methods needed and call the RTI class methods needed. However, most existing simulations will not have been designed to implement these methods and although the calls may be added, the simulations' concept of operation may not be integrated with the interface.

Compliancy assessment therefore needs to investigate how close a simulation is to the HLA concepts of operation. This means assessing how easy it is to:

- make specific object attributes available externally;
- identify where in the internal model hierarchy an incoming attribute / object change would be;
- trigger external methods whenever those attributes change;
- identify when interactions have occurred;
- process triggers indicating that interactions have occurred;
- dynamically change the entities being represented due to creation / deletion;
- split the modelling of object attributes to support attribute ownership transfer.

In many ways provision of these facilities is more valuable than the actual adherence to a full interface specification, since it would be very easy to write a model that adhered to the specification but did absolutely nothing. In effect we are trying to place a higher level behaviour specification on the federate and it is this that should be taken into account in the Compliancy assessment.

6.2 Recommendations for HLA interface documentation improvements

Having tried to understand the HLA interface both from the perspective of how the RTI behaves and how a federate should use it, the authors believe that the current documentation is missing some useful information. While we appreciate that in its early days the interface was intended to be non-prescriptive to encourage industry to discover the best ways of using it, there are certain constraints that must be met. These are due to the way the RTI itself has been implemented, and due to the way a federation intends operating.

As mentioned previously there tend to be a few distinct ways of operating simulations (DIS, ALSP, etc.) and showing how the current RTI would support these methods of simulation would rapidly present the interface in terms the user understands.

The object models demonstrated here would go a long way to providing an overview and index into the architecture and interface operations. The Use case diagrams provide the examples of behaviour that supply the missing information. Finally if the Select OMT Tool were available, and the interface documentation were released as a database, the interaction with the database allows concepts to be rapidly understood and imported directly into new federate designs.

6.3 Summary

This exercise has been both stimulating from an Intelligence Analysts' viewpoint and frustrating from an engineering viewpoint.

The omissions from the specification are a challenge to think about what it is you want to do, rather than what you are told to do. You have bits of information that make up part of the picture and then, to fill in the rest, you have to hypothesize or guess and then test the guess against the rest of the knowledge you have.

It is frustrating too since having spent significant effort deducing how the RTI must work and engineering a federate design to conform to this, a bit more information appears that would have made the whole job much easier.

As with most Intelligence information, it is only worthwhile if it is used. So rather than leave this study on a shelf somewhere and just read the next version of the HLA Specification, we offer our Intelligence Analysis as a basis for others to build on.

6.4 Acknowledgments

The authors wish to thank the War Gaming and Simulation Centre, CDA and Technology Group 10 of the UK MOD for sponsoring this work.

7. REFERENCES

1. http:\\www.dmso.mil
2. DMSO DOD High Level Architecture For Simulations Interface Specification.
3. DMSO DOD High Level Object Model Template
4. HLA Time Management: Design Document
5. Object-Oriented Modelling and Design -
J. Rumbaugh et al. ISBN 0-13-630054-5